APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/681,064	12/15/2000	Andrew L. Bliss	MSFT-0218	9482

41505   7590   06/14/2005

WOODCOCK WASHBURN LLP
ONE LIBERTY PLACE - 46TH FLOOR
PHILADELPHIA, PA 19103

EXAMINER
YIGDALL, MICHAEL J

ART UNIT	PAPER NUMBER
2192	

DATE MAILED: 06/14/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

<table>
<tr><td rowspan="2">Office Action Summary</td><td>Application No.
09/681,064</td><td>Applicant(s)
BLISS ET AL.</td></tr>
<tr><td>Examiner
Michael J. Yigdall</td><td>Art Unit
2192</td></tr>
</table>

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1) ☒ Responsive to communication(s) filed on *14 March 2005*.

2a) ☐ This action is **FINAL**.   2b) ☒ This action is non-final.

3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4) ☒ Claim(s) *1,3,6-11,14-20 and 24-31* is/are pending in the application.

   4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5) ☐ Claim(s) _____ is/are allowed.

6) ☒ Claim(s) *1,3,6-11,14-20 and 24-31* is/are rejected.

7) ☐ Claim(s) _____ is/are objected to.

8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9) ☐ The specification is objected to by the Examiner.

10) ☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

   Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

   Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   a)☐ All   b)☐ Some * c)☐ None of:

   1.☐ Certified copies of the priority documents have been received.

   2.☐ Certified copies of the priority documents have been received in Application No. _____.

   3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

   * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
   Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413)
   Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1.   A request for continued examination under 37 CFR 1.114, including the fee set forth in

37 CFR 1.17(e), was filed in this application after final rejection. Since this application is

eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e)

has been timely paid, the finality of the previous Office action has been withdrawn pursuant to

37 CFR 1.114. Applicant's submission filed on March 14, 2005 has been entered. Claims 1, 3,

6-11, 14-20 and 24-31 are pending.

Response to Arguments

2.   Applicant's arguments with respect to claims 1 and 20 have been considered but are moot

in view of the new ground(s) of rejection.

Applicant contends that the target memory address abstraction in FIG. 9 of You does not

even suggest the debugging type abstraction, especially inasmuch as You contains no suggestion

or hint that by providing such a debugging type abstraction, multiple types of debuggees can be

handled by a single debugger (Applicant's remarks, page 11, first paragraph).

However, You discloses a debugger portable to multiple operating systems and hardware

platforms (Applicant's remarks, page 10, third paragraph). The debugger is organized into a

reusable set of classes to provide debugging services that vary in implementation for each

debuggee (see, for example, column 66, lines 62-67). The classes form an abstraction that

includes common, shared programming code (see, for example, column 11, lines 23-25). One

such abstraction is a processor abstraction to support the memory addressing of each type of

processor (see, for example, column 27, lines 1-7). The programming code of this abstraction is

organized into a tree, with common code at the base node and derived code at the child nodes (see, for example, FIG. 9 and column 27, lines 13-24).

You further discloses a plurality of debugging types supported by the portable debugging services (see, for example, column 6, lines 31-55). Again, the architecture of the portable debugging services is organized into an abstraction of classes or "blocks" to support the plurality of debuggees and debugging types (see, for example, column 5, lines 43-55). You discloses that by providing such abstractions, multiple types of debuggees can be handled by a single debugger (see, for example, column 9, lines 17-25).

Therefore, You suggests a debugging type abstraction analogous to the processor abstraction for memory addresses illustrated in FIG. 9. Nonetheless, Niemi expressly discloses a debugging type abstraction, as presented in the claim rejections below.

Claim Rejections - 35 USC § 103

3.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4.   Claims 1, 3, 6-11, 14, 15, 18-20 and 24-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 6,158,045 to You (art of record, "You") in view of U.S. Pat. No. 6,470,388 to Niemi et al. (art made of record, "Niemi").

With respect to claim 1 (currently amended), You discloses a debugger for debugging any of a plurality of debuggees (see, for example, the abstract), each debuggee having a processor attribute selected from a plurality of processor attributes and representative of a type of processor associated with the debuggee (see, for example, column 72, line 55 to column 73, line 5, which shows that the debuggee has processor attributes from which the architecture or the type of the processor may be determined).

You does not expressly disclose each debuggee having a debugging type attribute selected from a plurality of debugging type attributes and representative of a type of debugging to be performed with respect to the debuggee.

However, You discloses that the engine is intended for a plurality of debugging types (see, for example, column 5, lines 43-55), and discloses a plurality of debugging types (see, for example, column 6, lines 31-55). It would have been obvious to one of ordinary skill in the art at the time the invention was made for each debuggee to include a debugging type attribute, in addition to the processor attributes (see, for example, column 9, lines 17-25), so as to designate a particular debugging type. Such an addition would, for example, enable the engine to perform the designated type of debugging without the need for user input.

You further discloses that the debugger is instantiated on a computer (see, for example, FIG. 1) and comprises:

(a) an engine for performing debugging functions with respect to any of the plurality of debuggees (see, for example, column 9, lines 17-25, which shows a portable debugging engine for debugging any of a plurality of debuggees), the engine including:

(i) a plurality of debugging type blocks (see, for example, column 66, lines 62-67, which shows that the engine includes a plurality of classes or blocks to provide types of debugging services), each debugging type block for supporting at least one of the plurality of debugging type attributes (see, for example, column 6, lines 31-55, which shows a plurality of debugging types supported by the engine); and

(ii) a plurality of processor blocks (see, for example, column 66, lines 62-67, which shows that the engine includes a plurality of classes or blocks to provide debugging services for a plurality of processors), each processor block for supporting at least one of the plurality of processor attributes (see, for example, column 9, lines 17-25, which shows a plurality of processor attributes supported by the engine),

(b) wherein a particular debugging type block and a particular processor block are selected for debugging a particular debuggee based on the debugging type attribute and processor attribute of the particular debuggee (see, for example, column 4, lines 41-50 and column 5, lines 47-59, which shows selecting the classes or blocks for debugging a particular debuggee based on attributes of the debuggee),

(c) wherein the plurality of debugging type blocks are organized into a debugging type abstraction available to provide debugging type services that vary in implementation for each debugging type (see, for example, see, for example, column 5, lines 43-59, which shows that the engine is organized into abstractions to provide debugging services that vary in implementation),

(d) wherein the debugging type abstraction comprises programming code, and wherein at least a portion of the programming code for the debugging type abstraction is common as between at least some debugging type blocks and is shared by such debugging type blocks (see,

for example, column 10, lines 29-40, which shows that the abstractions include common

programming code that is shared and reused), and

(e) wherein the programming code for the debugging type abstraction is organized into a

tree form with generic code at a base node and more specific levels of code branching out at

nodes therefrom, each debugging type block including at least one node from the tree (see, for

example, FIG. 9 and column 27, lines 13-24, which shows that the abstractions are organized

into trees with common code at the base node and derived code at the child nodes).

You illustrates a processor abstraction to provide memory addressing for each processor

(see, for example, FIG. 9 and column 27, lines 1-7), but does not expressly illustrate the

debugging type abstraction.

However, Niemi expressly discloses a debugging type abstraction as recited in the claim

that is organized into a tree and provides debugging type services (see, for example, FIG. 4 and

column 8, lines 11-42).

It would have been obvious to one of ordinary skill in the art at the time the invention

was made to supplement You with a debugging type abstraction, such as taught by Niemi, so as

to represent the debugging types supported by the engine (see, for example, You, column 5, lines

43-55), as intended (see, for example, You, column 5, lines 43-55).

With respect to claim 3 (currently amended), the rejection of claim 1 is incorporated, and

You further discloses the limitation wherein the debugging services include services selected

from a group consisting of accessing memory, accessing context, accessing system information,

inserting a breakpoint, removing a breakpoint, controlling execution, and combinations thereof

(see, for example, column 10, lines 20-27, which shows services such as stack or memory access, runtime or context information, breakpoints, and so on).

With respect to claim 6 (original), the rejection of claim 1 is incorporated, and You further discloses the limitation wherein the plurality of processor blocks are organized into a processor abstraction available to provide processor services that vary in implementation for each processor (see, for example, see, for example, column 5, lines 43-59, which shows that the engine is organized into abstractions to provide debugging services that vary in implementation, and see, for example, FIG. 9 and column 27, lines 1-7, which shows a processor abstraction to provide memory addressing for each processor).

With respect to claim 7 (original), the rejection of claim 6 is incorporated, and You further discloses the limitation wherein the processor services include services selected from a group consisting of recognizing particular processor instructions, recognizing processor states, maintaining hardware breakpoints, assembling code for the processor, disassembling code from the processor, disassembling code from a dump file produced by the processor, and combinations thereof (see, for example, column 10, lines 20-27, which shows services such as register or processor state information, breakpoints, hardware exceptions, and so on).

With respect to claim 8 (original), the rejection of claim 6 is incorporated, and You further discloses the limitation wherein the processor abstraction comprises programming code, and wherein at least a portion of the programming code for the processor abstraction is common as between at least some processor blocks and is shared by such processor blocks (see, for

example, column 10, lines 29-40, which shows that the abstractions include common

programming code that is shared and reused).

With respect to claim 9 (original), the rejection of claim 8 is incorporated, and You

further discloses the limitation wherein the programming code for the processor abstraction is

organized into a tree form with generic code at a base node and more specific levels of code

branching out at nodes therefrom, each processor block including at least one node from the tree

(see, for example, FIG. 9 and column 27, lines 13-24, which shows that the abstractions are

organized into trees with common code at the base node and derived code at the child nodes).

With respect to claim 10 (original), the rejection of claim 1 is incorporated, and You

further discloses the limitation wherein the engine further includes a high level portion for

issuing generic requests to the selected debugging type block and to the selected processor block

to accomplish debugging actions (see, for example, FIG. 2 and column 9, lines 28-45, which

shows the high-level debugging interface for issuing generic requests to the selected classes or

blocks of the engine).

With respect to claim 11 (original), the rejection of claim 10 is incorporated, and You

further discloses the limitation wherein the plurality of debugging type blocks are organized into

a debugging type abstraction available to provide debugging type services that vary in

implementation for each debugging type, wherein the plurality of processor blocks are organized

into a processor abstraction available to provide processor services that vary in implementation

for each processor, and wherein the high level portion issues generic request to the debugging

type abstraction and to the processor abstraction to accomplish debugging actions (see, for

example, see, for example, column 5, lines 43-59, which shows that the engine is organized into

abstractions to provide debugging services that vary in implementation, and see, for example,

FIG. 2 and column 9, lines 28-45, which shows the high-level debugging interface for issuing

generic requests to the selected classes or blocks of the engine).

With respect to claim 14 (original), the rejection of claim 1 is incorporated, and You

further discloses the limitation wherein the plurality of processor attributes supported by the

processor blocks include processor attributes representative of members selected from a group

consisting of an X86 processor family, an ALPHA processor family, and IA64 processor family,

and combinations thereof (see, for example, FIG. 9, which shows support for X86 and 64-bit

processor families, among others).

With respect to claim 15 (original), the rejection of claim 1 is incorporated, and You

further discloses the limitation wherein the debugger further has an executable for being

executed by a user, for calling the engine, and for providing an interface between the user and

the engine (see, for example, column 9, lines 28-45, which shows the client interface executed by

a user for calling the engine).

With respect to claim 18 (original), the rejection of claim 1 is incorporated, and You

further discloses the limitation wherein the particular debuggee is a dump file produced by a

processor operating in a particular mode, wherein the debugging type attribute of the dump file

corresponds to the particular mode, and wherein the particular debugging type block of the

engine selected for debugging the dump file supports the debugging type attribute of the dump

file (see, for example, column 6, lines 31-55, which shows the debugging types supported by the

engine, including postmortem debugging for inspecting the state of a program after it terminates,

such as with a dump file that would identify the processing mode, such as with an attribute).

With respect to claim 19 (original), the rejection of claim 1 is incorporated, and You

further discloses the limitation wherein the particular debuggee is a dump file produced by a type

of processor, wherein the processor attribute of the dump file corresponds to the type of

processor, and wherein the particular processor block of the engine selected for debugging the

dump file supports the processor attribute of the dump file (see, for example, column 6, lines 31-

55, which shows the debugging types supported by the engine, including postmortem debugging

for inspecting the state of a program after it terminates, such as with a dump file that would

identify the type of processor, such as with an attribute, and see, for example, column 9, lines 17-

25, which shows a plurality of processor attributes supported by the engine).

With respect to claim 20 (currently amended), the limitations recited in the claim are

analogous to those of claim 1 (see the rejection of claim 1 above).

With respect to claim 24 (original), the limitations recited in the claim are analogous to

those of claim 6 (see the rejection of claim 6 above).

With respect to claim 25 (original), the limitations recited in the claim are analogous to

those of claim 8 (see the rejection of claim 8 above).

With respect to claim 26 (original), the limitations recited in the claim are analogous to

those of claim 9 (see the rejection of claim 9 above).

With respect to claim 27 (original), the limitations recited in the claim are analogous to those of claim 10 (see the rejection of claim 10 above).

With respect to claim 28 (original), the limitations recited in the claim are analogous to those of claim 11 (see the rejection of claim 11 above).

With respect to claim 29 (original), the limitations recited in the claim are analogous to those of claim 15 (see the rejection of claim 15 above).

5.   Claims 16, 17, 30 and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over You in view of Niemi, as applied to claims 15 and 29 above, respectively, and further in view of U.S. Pat. No. 5,533,192 to Hawley et al. (art of record, "Hawley").

With respect to claim 16 (original), the rejection of claim 15 is incorporated, but although You discloses support for a plurality of debugging types (see, for example, column 6, lines 31-55), You does not expressly disclose the limitation wherein the executable includes an attribute that results in the selection of a particular debugging type block in the engine.

However, Hawley discloses an attribute in the executable used to select a particular debugger (see, for example, information 811 in FIG. 8A, and see, for example, step 853 in FIG. 8B and column 19, lines 12-22), in a debugging system having a plurality of debuggers (see, for example, the abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement You with an attribute to select the appropriate debugger or debugging

type block, such as taught by Hawley, in the absence of an alternative selection by the user (see, for example, Hawley, column 18, lines 26-37).

With respect to claim 17 (original), the rejection of claim 15 is incorporated, but although You discloses support for a plurality of processor families or types (see, for example, column 9, lines 17-26), You does not expressly disclose the limitation wherein the executable includes an attribute that results in the selection of a particular processor block in the engine.

However, Hawley discloses an attribute in the executable used to select a particular debugger (see, for example, information 811 in FIG. 8A, and see, for example, step 853 in FIG. 8B and column 19, lines 12-22), in a debugging system having a plurality of debuggers (see, for example, the abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to supplement You with an attribute to select the appropriate debugger or processor block, such as taught by Hawley, in the absence of an alternative selection by the user (see, for example, Hawley, column 18, lines 26-37).

With respect to claim 30 (original), the limitations recited in the claim are analogous to those of claim 16 (see the rejection of claim 16 above).

With respect to claim 31 (original), the limitations recited in the claim are analogous to those of claim 17 (see the rejection of claim 17 above).

Conclusion

6.   The prior art made of record and not relied upon is considered pertinent to Applicant's

disclosure. U.S. Pat. No. 6,430,705 to Wisor et al. discloses a method for utilizing virtual

hardware descriptions to allow for multi-processor debugging in environments using varying

processor revision levels.

7.   Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Michael J. Yigdall whose telephone number is (571) 272-3707.

The examiner can normally be reached on Monday through Friday from 7:30am to 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the

organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PAIR) system. Status information for published applications

may be obtained from either Private PAIR or Public PAIR. Status information for unpublished

applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR

system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

MY

Michael J. Yigdall
Examiner
Art Unit 2192

mjy

TUAN DAM
SUPERVISORY PATENT EXAMINER